UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/755,502	01/05/2001	Arthur H. Khu	X-779 US	5243

24309 7590 05/17/2004

XILINX, INC
ATTN: LEGAL DEPARTMENT
2100 LOGIC DR
SAN JOSE, CA 95124

EXAMINER	
YIGDALL, MICHAEL J	

ART UNIT	PAPER NUMBER
2122	4

DATE MAILED: 05/17/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)
	09/755,502	KHU, ARTHUR H.
	Examiner	Art Unit
	Michael J. Yigdall	2122

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *09 March 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-27* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-27* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This Office action is in reply to Applicant's response and amendment dated March 9, 2004. Claims 1-27 are now pending and have been examined.

Response to Arguments

2. Applicant's arguments with respect to claims 1-20 have been considered but are moot in view of the new ground(s) of rejection.

Claim Rejections - 35 USC § 112

3. The rejection of claim 8 under 35 U.S.C. 112, first paragraph, is withdrawn in view of Applicant's remarks. Likewise, the rejection of claim 8 under 35 U.S.C. 112, second paragraph, is withdrawn in view of Applicant's remarks.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-27 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Enhanced Code Compression for Embedded RISC Processors" by Cooper et al. (hereinafter "Cooper").

With respect to claim 1, Cooper discloses a method of optimizing computer program code where the computer program code includes a plurality of statements (see page 139, the abstract), the method comprising the steps of:

(a) identifying a keyword statement;

(b) searching the program code for the keyword statement; and

(c) determining if the keyword statement begins a repeating pattern of statements in the program code.

Cooper discloses steps (a), (b) and (c) above in terms of finding repeated patterns in the program code by identifying each instruction or keyword statement and searching for equivalent sets of instructions (see page 140, sections 2 and 2.1).

Although Cooper discloses replacing a repeated pattern of statements with a jump instruction equivalent to the repeated pattern (see page 141, section 3 and Figure 4), Cooper does not expressly disclose the step of:

(d) replacing the repeating pattern of statements with a program loop equivalent to the repeating pattern of statements.

However, one of ordinary skill in the art can appreciate that if the regions of repeated instructions occurred in succession, replacing a set of repeated instructions with a jump instruction would, in effect, create a program loop. Moreover, a program loop is simply a set of instructions followed by a jump instruction that returns to the start of the repeated pattern.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made that, when a repeated pattern of statements occurs a number of times in

succession, replacing the repeated pattern with a jump instruction would have the effect of replacing the repeated pattern with an equivalent program loop.

With respect to claim 2, Cooper further discloses the limitation wherein the keyword statement includes a keyword and an optional data reference, and prior to the searching step:

(a) sequentially locating each keyword statement in the program code; and

(b) converting the optional data reference, if present, from each located keyword statement to a data array reference.

Cooper discloses the limitation and the steps above in terms of locating each instruction or keyword statement, which may include data references such as registers and constants, and converting the instruction and its references to an entry in a table or data array (see page 140, right-hand column, first paragraph).

With respect to claim 3, Cooper further discloses the limitation wherein the converting includes assigning an array index value to the data array reference where each located keyword statement is assigned a next sequential value of the array index value (see page 140, right-hand column, second paragraph, which shows that each instruction or keyword statement was assigned an index value into the table or data array).

With respect to claim 4, Cooper further discloses the limitation wherein the determining step further includes:

(a) comparing data array references of two keyword statements from the program code; and

(b) determining if the array index values from the data array references match in size and sequential order.

Coopers discloses the steps above in terms of comparing the entries of two instructions or keyword statements in the table or data array to determine if the two instructions match, i.e. in size and in order, by hashing the instructions and obtaining the index values (see page 140, right-hand column, first and second paragraphs).

With respect to claim 5, Cooper further discloses the limitation wherein the determining step includes:

(a) determining a first pattern of statements in the program code beginning with a first keyword statement and ending with a statement preceding a second keyword statement that sequentially appears in the program code after the first keyword statement;

(b) determining a second pattern of statements in the program code beginning with the second keyword statement and ending with a statement preceding a third keyword statement that sequentially appears in the program code after the second keyword statement; and

(c) comparing the first pattern of statements to the second pattern of statements; and

(d) setting the first pattern of statements as a repeating pattern if the first and second pattern of statements substantially match.

Cooper discloses the steps above in terms of finding repeated patterns in the program code, which are delineated by first, second, third, etc. instructions or keyword statements, by comparing sets of instructions and determining whether the sets are equivalent (see page 140, sections 2 and 2.1; see also page 140, right-hand column, third paragraph and Figure 2, which show two matching patterns of statements, i.e. a repeated pattern, in the program code).

With respect to claim 6, although Cooper discloses replacing a repeated pattern of statements with a jump instruction (see page 141, section 3 and Figure 4), Cooper does not expressly disclose the limitation wherein the replacing step includes:

(a) generating loop code for executing a loop within the source code at location of the repeating pattern of statements;

(b) inserting one instance of the repeating pattern of statements within the loop code; and

(c) defining the loop code iterate a number of times equal to a number of instances of the repeating pattern.

However, as set forth above for claim 1, it would have been obvious to one of ordinary skill in the art at the time the invention was made that, when a repeated pattern of statements occurs a number of times in succession, replacing the repeated pattern with a jump instruction would have the effect of replacing the repeated pattern with an equivalent program loop.

The jump instruction, which may, in effect, generate a loop, is inserted at the location of the repeated pattern, as in step (a) above (see page 141, section 3 and Figure 4), leaving one instance of the repeated pattern in the program code, as in step (b) above (see page 140, right-hand column, third paragraph). The repeated pattern would iterate the same number of times as the number of instances of the repeated pattern in the original code, as in step (c) above.

With respect to claim 7, Cooper further discloses the limitation wherein the keyword statement is identified from a predetermined keyword statement (see page 140, right-hand column, first paragraph, which shows that the instruction or keyword statement is identified based on the opcode, i.e. a predetermined keyword statement from the instruction set).

With respect to claim 8, Cooper further discloses the limitation wherein the keyword statement is identified from a selection made by a user (see page 140, right-hand column, first paragraph, which shows that the instruction or keyword statement is identified based on the specified registers and constants, i.e. selections made by the user).

With respect to claim 9, Cooper further discloses identifying a plurality of keyword statements and repeating the method for optimizing for each of the plurality keyword statements (see page 140, sections 2 and 2.1, which shows identifying each instruction or keyword statement and finding all of the repeats in the program).

With respect to claim 10, Cooper discloses a software code optimizer (see page 139, the abstract) comprising:

(a) analyzing program instructions for analyzing a software code and determining an occurrence of a repeating pattern of code therein (see page 140, sections 2 and 2.1, which show analyzing software code to find repeated patterns of instructions).

Although Cooper discloses replacing a repeated pattern of statements with a jump instruction to perform an equivalent function as the original code (see page 141, section 3 and Figure 4), Cooper does not expressly disclose:

(b) converting program instructions for converting the repeating pattern of code to a programming loop that performs an equivalent function as the repeating pattern of code.

However, one of ordinary skill in the art can appreciate that if the regions of repeated instructions occurred in succession, replacing a set of repeated instructions with a jump

instruction would, in effect, create a program loop. Moreover, a program loop is simply a set of instructions followed by a jump instruction that returns to the start of the repeated pattern.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made that, when a repeated pattern of statements occurs a number of times in succession, replacing the repeated pattern with a jump instruction would have the effect of converting the repeated pattern to an equivalent program loop.

With respect to claim 11, Cooper further discloses the limitation wherein the analyzing program instructions further include:

(a) program instructions for searching the software code for a keyword; and

(b) program instructions for identifying if the keyword begins a repeating pattern of code within the software code and determining a number of occurrences the repeating pattern.

Cooper discloses the features above in terms of identifying repeated patterns of code by searching for instructions or keywords (see page 140, sections 2 and 2.1), and determining the number of fragments or occurrences of the repeated pattern (see page 140, right-hand column, third paragraph).

With respect to claim 12, Cooper further discloses program instructions for repeating the analyzing program instructions for a plurality of keywords (see page 140, sections 2 and 2.1, which shows analyzing each instruction or keyword in the program).

With respect to claim 13, although Cooper discloses replacing a repeated pattern of statements with a jump instruction (see page 141, section 3 and Figure 4), Cooper does not expressly disclose the limitation wherein the converting program instructions further include

program instructions for setting the programming loop to repeat a number of times equal to the number of occurrences of the repeating pattern and inserting one occurrence of the repeating pattern within the programming loop.

However, as set forth above for claim 10, it would have been obvious to one of ordinary skill in the art at the time the invention was made that, when a repeated pattern of statements occurs a number of times in succession, replacing the repeated pattern with a jump instruction would have the effect of converting the repeated pattern to an equivalent program loop.

One occurrence of the repeated pattern is left within the program code (see page 140, right-hand column, third paragraph), and the repeated pattern would iterate the same number of times as the number of original occurrences (see page 141, section 3 and Figure 4).

With respect to claim 14, Cooper further discloses program instructions for locating data references in each statement of program code containing the keyword and converting the data references to data array references (see page 140, right-hand column, first paragraph, which shows locating each instruction or keyword statement, which may include data references such as registers and constants, and converting the instruction and its references to an entry in a table or data array).

With respect to claim 15, Cooper further discloses a compiler for translating the software code to an object code executable by a computer (see page 139, the abstract, which shows an optimizing compiler for translating the software code to executable code).

With respect to claim 16, Cooper discloses a process for optimizing a software code that includes a plurality of statements (see page 139, the abstract), the process comprising the steps of:

(a) locating multiple occurrences of a code pattern within the software code where the multiple occurrences appear sequentially to each other in the software code (see page 140, sections 2 and 2.1, which show locating repeated patterns of instructions in the software code, i.e. including multiple occurrences appearing sequentially).

Although Cooper discloses replacing a repeated pattern of statements with a jump instruction to produce an equivalent result as the original code (see page 141, section 3 and Figure 4), Cooper does not expressly disclose the steps of:

(b) generating a program loop that executes one occurrence of the code pattern a number of times to produce an equivalent result as executing the multiple occurrences of the code pattern; and

(c) replacing the multiple occurrences the code pattern the software code with the program loop.

However, one of ordinary skill in the art can appreciate that if the regions of repeated instructions occurred in succession, replacing a set of repeated instructions with a jump instruction would, in effect, create a program loop. Moreover, a program loop is simply a set of instructions followed by a jump instruction that returns to the start of the repeated pattern.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made that, when a repeated pattern of statements occurs a number of times in succession, replacing the repeated pattern with a jump instruction would have the effect of replacing the repeated pattern with an equivalent program loop.

With respect to claim 17, Cooper further discloses:

(a) selecting a keyword statement; and

(b) defining the code pattern based on the keyword statement.

Cooper discloses the steps above in terms of defining repeated patterns in the program code by analyzing each instruction or keyword statement (see page 140, sections 2 and 2.1).

With respect to claim 18, Cooper further discloses the limitation wherein the defining step includes:

(a) locating a first instance of the keyword statement in the software code;

(b) defining a first code pattern to include at least the first instance of the keyword statement;

(c) adding subsequent non-keyword statements to the first code pattern until a second instance of the keyword statement appears in the software code;

(d) defining a second code pattern to include at least the second instance of the keyword statement;

(e) adding subsequent non-keyword statements to the second code pattern until a third instance of the keyword statement appears in the software code or until a number of the subsequent non-keyword statements added equal a number of the subsequent non-keyword statements in the first code pattern; and

(f) comparing the first code pattern with the second code pattern to determine if the second code pattern is a multiple occurrence the first code pattern.

Cooper discloses the steps above in terms of finding repeated patterns in the software code, which are delineated by first, second, third, etc. instances of instructions or keyword statements, by comparing sets of instructions and determining whether the sets are equivalent (see page 140, sections 2 and 2.1, which also shows constructing a suffix tree by adding

statements and forming nodes or patterns; see also page 140, right-hand column, third paragraph and Figure 2, which show two matching patterns of statements, i.e. a multiple occurrences of a pattern, in the program code).

With respect to claim 19, Cooper further discloses, prior to the locating step:

(a) selecting at least one keyword statement where the keyword statement includes a keyword and an optional data reference; and

(b) converting each of the data references that appear in each keyword statement in the software code to a data array reference, the data array reference being loaded with values of the converted data references.

Cooper discloses the limitation and the steps above in terms of locating each instruction or keyword statement, which may include data references such as registers and constants, and converting the instruction and its references to an entry in a table or data array (see page 140, right-hand column, first paragraph).

With respect to claim 20, Cooper further discloses the limitation wherein the generating a program loop step includes generating a looping instruction (see page 141, section 3 and Figure 4, which shows replacing a repeated pattern of statements with a jump instruction; as set forth above for claim 16, a looping instruction is, in effect, a jump instruction).

With respect to new claims 21-27, see the explanations regarding Cooper set forth above for claims 1-20. The steps and features of the method and apparatus recited in claims 21-27 are analogous to the limitations recited in the preceding claims. The new claims are purportedly

directed to the same invention, recited in alternative language (see Applicant's remarks, page 9, first section).

Conclusion

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (703) 305-0352. The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2122

mjy
May 7, 2004

TUAN DAM
SUPERVISORY PATENT EXAMINER